1. INTRODUCTION

1.1 Problem Statement

To construct a decision tree using ID3 algorithm using real data implemented in any programming language.

1.2 Decision Tree

In simple words, a decision tree is a structure that contains nodes and edges and is built from a dataset. Each node is either used to ***make a decision*** *(*known as decision node) or to ***represent an outcome*** (known as leaf node).

1.2 ID3 Algorithm

ID3 stands for Iterative Dichotomiser 3 and is named such because the algorithm iteratively (repeatedly) dichotomizes(divides) features into two or more groups at each step.

ID3 uses a **top-down greedy** approach to build a decision tree. In simple words, the **top-down** approach means that we start building the tree from the top and the **greedy** approach means that at each iteration we select the best feature at the present moment to create a node. Most generally ID3 is only used for classification problems with *nominal* features only.

2. ALGORITHM

2.1 Entropy

Entropy is a measure of purity of a dataset. Given a training dataset D with two classes, P and S, E(D) or entropy of dataset is calculated as-
Let *p* = number of *P*/number of records, *s* = number of *S*/number of records

$$E(D) = -p \log_2 p - s \log_2 s$$

General formula-

$$E(D) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

If all records belong to the same class, entropy is 0 else if it is equally distributed over all classes, entropy is 1.

2.2 Information Gain

Information Gain measures the effectiveness of an attribute in classification. Information Gain of an attribute is the expected reduction in entropy caused by partitioning the collection according to this attribute.

$$Gain(D,V) = E(D) - \frac{Dv}{D} \; x \; E(D_v)$$

It is a measurement of a splitting called Information Gain to determine the goodness of a split.

2.3 ID3 Algorithm

1) Establish Classification Attribute in the dataset D.

2) Compute Classification Entropy.

3) For each attribute in D, calculate Information Gain using classification attribute.

4) Select Attribute with the highest Information Gain to be the next Node in the tree (starting from the Root node).

5) Divide the records into classes based on the selected Node Attribute.

6) Remove Node Attribute from D, creating reduced dataset D_S.

7) Repeat steps 3-6 until all attributes have been used, or the same classification value remains for all rows in the reduced dataset.

8) Construct the final decision tree.

3. DATASET

In this project, **Graduate-Admission-Prediction** dataset has been used which has been taken from Kaggle: (https://www.kaggle.com/mohansacharya/graduate-admissions/home).
The dataset predicts the chances of admission of a student given his/her scores information. There are 400 entries in the dataset and the decision is made based on 7 attributes as shown below. The last column represents the binary decision of acceptance or rejection.

Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
1.	337	118	4	4.5	4.5	9.65	1	0.92
2.	324	107	4	4	4.5	8.87	1	0.76
3.	316	104	3	3	3.5	8	1	0.72
4.	322	110	3	3.5	2.5	8.67	1	0.80
5.	314	103	2	2	3	8.21	0	0.65

Table 1

The above dataset (table 1) is converted into a dataset as shown below (table 2) such that the numerical values are converted into their corresponding categories so that we can achieve a fully nominal dataset. This is done for the ease of calculations. The criteria for the conversion is subject to general trends and in this particular case, it has been derived from gathering local information and analyzing the data and the relationships among the attributes.

The conversion rules used are as follows-

a) GRE SCORE- 'GRE_Good' if >=310 else if <290 'GRE_Poor' else 'GRE_Med'

b) TOEFL SCORE- 'TOEFL_Good' if >=110 else 'TOEFL_Poor'

c) University Rating- 'UNI_Good' if >=4.0 else 'UNI_Poor'

d) SOP- 'SOP_Good' if >=4.0 else 'SOP_Poor'

e) LOR- 'LOR_Good' if >=4.0 else 'LOR_Poor'

f) CGPA- 'CGPA_Good' if >=8.5 else if <7.0 'CGPA_Poor' else 'CGPA_Med'

g) Research- 'Research_Yes' if ==1 else 'Research_No'

h) Chance of Admit- 'Admit_Yes' if >=0.50 else 'Admit_No'

Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
1.	GRE_ Good	TOEFL_ Good	UNI_Good	SOP_G ood	LOR_G ood	CGPA_ Good	Research _Yes	Admit_Ye s
2.	GRE_ Good	TOEFL_ Poor	UNI_Good	SOP_G ood	LOR_G ood	CGPA_ Good	Research _Yes	Admit_Ye s
3.	GRE_ Good	TOEFL_ Poor	UNI_Poor	SOP_Po or	LOR_Po or	CGPA_ Med	Research _Yes	Admit_Ye s
4.	GRE_ Good	TOEFL_ Good	UNI_Poor	SOP_Po or	LOR_Po or	CGPA_ Good	Research _Yes	Admit_Ye s
5.	GRE_ Good	TOEFL_ Poor	UNI_Poor	SOP_Po or	LOR_Po or	CGPA_ Med	Research _No	Admit_Ye s

Table 2

4. METHODOLOGY

1. The code is implemented in python.
2. The dataset is imported by pandas and modified to represent nominal values.
3. The list *rows* stores the indices of all the records and list *columns* stores the indices of all the attributes.
4. A *Node* class is created containing value, decision and child as its components.
5. *Calculate()* function is called on the dataset which further calls the *buildTree()* function to build the decision tree which creates the required nodes.
6. It further calls *findMaxGain()* function to calculate information gain for all attributes and returns the attribute with maximum gain.
7. *findEntropy()* function is called to calculate the entropy.
8. *Traverse*() function prints the required preorder traversal.

5. RESULTS

The preorder traversal of the decision tree is as shown-

Start	GRE Score	GRE_Good	TOEFL Score	TOEFL_Good	Admit_Yes
TOEFL_Poor	LOR	LOR_Good	Admit_Yes	LOR_Poor	SOP
SOP_Poor	Research	Research_Yes	CGPA	CGPA_Med	University Rating
UNI_Poor	Admit_No	CGPA_Good	Admit_Yes	Research_No	University Rating
UNI_Poor	CGPA	CGPA_Med	Admit_No	CGPA_Good	Admit_No

UNI_Good	Admit_Yes	SOP_Good	Admit_Yes	GRE_Med	LOR
LOR_Good	Admit_Yes	LOR_Poor	CGPA	CGPA_Med	SOP
SOP_Poor	TOEFL Score	TOEFL_Poor	University Rating	UNI_Poor	Research
Research_No	Admit_No	Research_Yes	Admit_No	UNI_Good	Research
Research_No	Admit_No	TOEFL_Good	Admit_No	SOP_Good	Research
Research_Yes	Admit_Yes	Research_No	Admit_No	CGPA_Good	Admit_Yes
CGPA_Poor	Admit_No	GRE_Poor	Admit_No		

The resultant preorder traversal has 70 nodes, consisting of a **start node** *(Start)*, **attribute nodes** *(CGPA, SOP, LOR, Research etc)*, **attribute value nodes** *(SOP_Good, Research_Yes, CGPA_Med etc)*, and the **decision nodes** *(Admit_Yes or Admit_No)*.

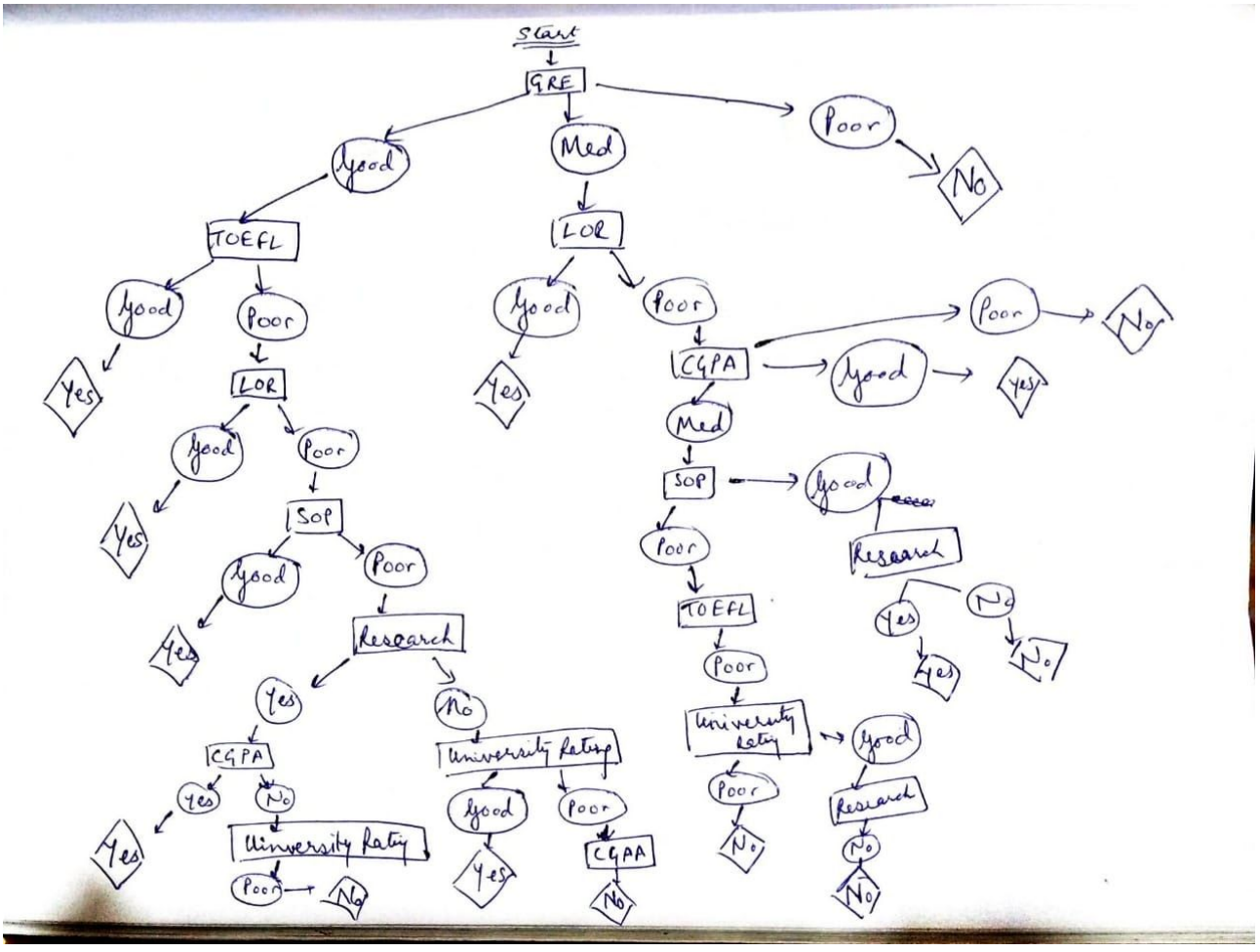

Figure 1

The above figure shows partial result of the preorder traversal into a decision tree. The circular nodes represent attribute values, rectangular nodes represent attributes, the diamond notes represent final decision (admit_yes or admit_no).